

On this page



Dockerfile

In the previous step, you ran a command that prints the container's uptime. Now you'll create a Dockerfile to customize the contents of your own Docker image.

Create a Dockerfile

Create a new file called `Dockerfile` and add the following items.

```
FROM busybox
COPY entrypoint.sh /
RUN chmod +x /entrypoint.sh
ENTRYPOINT ["/entrypoint.sh"]
```

This Dockerfile starts from the `busybox` image like we used before. It then adds a custom `entrypoint.sh` script, makes it executable, and configures it as the entrypoint.

The entrypoint script

Now let's create `entrypoint.sh` with the following contents:

```
#!/bin/sh
echo "The time is: $(date)"
```

NOTE

While we named this script `entrypoint.sh` you will see a variety of naming conventions; such as:

- `start.sh`
- `CMD.sh`
- `entry_path.sh`

**Ask AI**

These files are normally placed in a folder called `script` but it is dependent on the maintainers of that repository.

This is a simple script that will print the current time when the container starts.

Why an entrypoint script:

- It lets you customize what command gets run when a container starts from your image.
- For example, our script runs `date` to print the time.
- Without it, containers would exit immediately after starting.
- Entrypoints make images executable and easier to reuse.

Build the image

With those files created, we can now build a Docker image using our Dockerfile:

```
docker image build -t my-time-image .
```

This will build the image named `my-time-image` from the Dockerfile in the current directory.

Why build a custom image:

- Lets you package up custom dependencies and configurations.
- For example you can install extra software needed for your app.
- Makes deploying applications more reliable and portable.
- Instead of installing things manually on every server, just use your image.
- Custom images can be shared and reused easily across environments.
- Building images puts your application into a standardized unit that "runs anywhere".
- You can version images over time as you update configurations.

Run the image

Finally, let's run a container from our new image:

```
docker run my-time-image
```




Ask AI

We should see the same output as before printing the current time!

Entrypoints and Dockerfiles let you define reusable, executable containers that run the software and commands you need. This makes deploying and sharing applications much easier without per-server configuration.

By putting commands like this into a Dockerfile, you can easily build reusable and shareable images.

 [Edit this page](#)

Previous

« [Containers overview](#)

Next

[Persist data outside of containers](#) »

Docs

[Overview](#)

[Tutorials](#)

[AI APIs](#)

Community

[Discord](#) 

[Contact us](#) 

More

[Blog](#) 

[GitHub](#) 

